REMARKS

Claims 1-31 are pending. Claims 1-3, 11-16, 20-23, and 31 were rejected under 35 U.S.C. 102(e) as being anticipated by Frankel (USPAP 20030093254). In the previous Office Action, claims 4-10, 17-19, and 24-29 were objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including base and intervening claim limitations. Independent claims 1, 20, and 30, were amended to incorporate the allowable dependent and intervening limitations to facilitate prosecution.

However, in the Advisory Action, the Examiner is now stating that the previously allowable limitation is now potentially rejectable by the same reference Frankel. More specifically, the Examiner is arguing that the recitation "wherein means for generating hardware acceleration logic comprises means for identifying pointer access in the portion of the high-level language program" is now "potentially rejectable" by Frankel paragraphs 37, 40, and 41.

Frankel notes that "The grammar may include one or more commands for defining the configuration of the system under test. In one embodiment, these commands include a port of view (POV) command, a device description file (DDF) command, and a system configuration file (SCF) command. These commands may, in one implementation, be stored as files rather than message packets transmitted between nodes in the distributed simulation system. However, these commands are part of the grammar and may be transmitted as message packets if desired." (paragraph 0037)

Furthermore, "the POV command defines the logical port types for the system under test. Generally, signal information (which includes at least a signal value, and may optionally include a strength for the signal) is transmitted through a logical port in a message packet. That is, a message packet which is transmitting signal information transmits the signal information for one or more logical ports of a port type defined in the POV command. Accordingly, the POV command specifies the format of the signal transmission message packets. Generally, a logical port is an abstract representation of one or more physical signals. For example, the set of signals which comprises a particular interface (e.g. a predefined bus interface, a test interface, etc.) may be grouped together into a logical port. Transmitting a set of values grouped as a logical port may more easily indicate to a user that a communication is occurring on the particular interface than if the physical signals are transmitted with values." (paragraph 0038)

"In one embodiment, the logical ports may be hierarchical in nature. In other words, a given logical port may contain other logical ports. Accordingly, multiple levels of abstraction may be defined, as desired. For example, a bus interface which is pipelined, such that signals are used at different phases in a transaction on the bus interface (e.g. arbitration phase, address phase, response phase, etc.) may be grouped into logical ports for each phase, and the logical ports for the phases may be grouped into a higher level logical port for the bus as a whole. Specifically, in one embodiment, a logical port comprises at least one logical port or logical signal, and may comprise zero or more logical ports and zero or more logical signals in general. Both the logical ports and the logical signals are defined in the POV command. It is noted that the term "port" may be used below instead of "logical port". The term "port" is intended to mean logical port in such contexts." (paragraph 0039)

Frankel, however, does not teach or suggest ""wherein means for generating hardware acceleration logic comprises means for identifying pointer access in the portion of the high-level language program." Frankel does not even mention pointers. Furthermore, Frankel does not even mention ideas associated with pointers, ideas such as memory addresses or memory access. Frankel only describes particular commands for defining port types.

"The DDF command is used to map logical signals (defined in the POV command) to the physical signals which appear in the models of the components of the system under test. In one embodiment, there may be at least one DDF command for each component in the system under test." (paragraph 0040) "The SCF command is used to instantiate the components of the system under test and to connect logical ports of the components of the system under test. The SCF command may be used by the hub for routing signal transmission message packets from one node to another." (paragraph 0041)

By contrast, according to particular embodiments of the present invention, it is recognized that conventional "high-level language tools have very poor pointer support. Read and write accesses to specific memory addresses conventionally are not easily implemented in hardware. Although a central processing unit (CPU) may have access to a specific address 0xFF3823, hardware accelerators usually do not have access the memory lines that the CPU has access to. Hardware accelerators typically have access to only a portion of memory, e.g. 0x00FF00 to 0x010000. A hardware accelerator can not be easily configured to access a memory

line outside of its allocated address space. Consequently, to support hardware acceleration code involving pointers, complex sequences of memory copies are used. Repeatedly copying various memory lines can be extremely inefficient and often eliminates the advantage of using hardware acceleration in the first place. Consequently, the techniques of the present invention allow the conversion of portions of high-level language programs into hardware without requiring any modifications to the underlying program. Pointer referencing and dereferencing is robust, while being handled automatically without user intervention. The techniques of the present invention allow the implementation of high-level language programs onto a variety of devices." (page 8, lines 5 – 22)

Furthermore, the Examiner argued that claim 31 is a substantial copy of original claim 1. The Applicants respectfully disagree. Claim 31 recites "wherein the portion is identified automatically using profiling data." Nothing cited by the Examiner is believed to teach or suggest a portion of a high-level language automatically identified using profiling data. Although claim 31 is believed allowable in its present form, claim 31 has been amended to facilitate prosecution. Claim 31 now recites wherein generating hardware acceleration logic includes pointer referencing and pointer dereferencing. Nothing cited by the Examiner teaches or suggests "wherein generating hardware acceleration logic includes pointer referencing and pointer dereferencing," pointers, memory address, or memory access.

All claims are now believed allowable. Applicants believe that all pending claims are allowable in their present form. Please feel free to contact the undersigned at the number provided below if there are any questions, concerns, or remaining issues.

Respectfully submitted,
BEYER WEAVER LLP

Godfrey K. Kwan
Reg. No. 46,850

P.O. Box 70250
Oakland, CA 94612-0250
(510) 663-1100